Attorney Docket No. 15437-0508 (P5373)

Patent

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:

Ron Karim

Serial No.: 09/880,231

Filed: June 12, 2001

Confirmation No.: 5058

Group Art Unit No.: 2126

Examiner: Qing Yuan Wu

For: MECHANISM FOR SAFELY EXECUTING AN UNTRUSTED PROGRAM

Mail Stop Amendment
Commissioner of Patents
P.O. Box 1450
Alexandria, VA 22313-1450

RESPONSE PURSUANT TO 37 C.F.R. § 1.116

Sir:

This is in response to the Final Office Action mailed February 11, 2005, the shortened statutory period for which runs until May 11, 2005.

Amendments to the claims and Remarks are presented on separate sheets below.

AMENDMENTS TO THE CLAIMS

1. (CURRENTLY AMENDED) A computer-implemented method for executing an untrusted program, comprising:

establishing a limited environment within a general environment, wherein said limited environment comprises ~~at least~~ one or more mock resources ~~resource~~, wherein said general environment comprises ~~at least~~ one or more real resources ~~resource~~, ~~and~~ wherein said limited environment and said general environment are both ~~implemented using~~ provided by the same ~~type of~~ operating system, and wherein programs executing within said limited environment cannot access the one or more real resources in said general environment;

executing at least a portion of an untrusted program within said limited environment; and

examining said limited environment after execution of at least said portion of said untrusted program to check for undesirable behavior exhibited by said untrusted program.

2. (CANCELLED).

3. (ORIGINAL) The method of claim 1, wherein said limited environment comprises a shell in a UNIX operating system environment.

4. (CURRENTLY AMENDED) The method of claim 1, wherein examining said limited ~~mock~~ environment comprises:

determining whether ~~said~~ a particular mock resource of said one or more mock resources has been deleted.

5. (CURRENTLY AMENDED) The method of claim 1, wherein examining said limited ~~mock~~ environment comprises:

determining whether ~~said~~ a particular mock resource of said one or more mock resources has been renamed.

6. (CURRENTLY AMENDED) The method of claim 1, wherein examining said limited ~~mock~~ environment comprises:

determining whether ~~said~~ a particular mock resource of said one or more mock resources has been moved.

7. (CURRENTLY AMENDED) The method of claim 1, wherein examining said limited ~~mock~~ environment comprises:

determining whether ~~said~~ a particular mock resource of said one or more mock resources has been altered.

8. (CURRENTLY AMENDED) The method of claim 7, wherein said particular mock resource has a parameter associated therewith which changes when said particular mock resource is altered, and wherein determining whether said particular mock resource has been altered, comprises:

determining whether said parameter has changed.

9. (CURRENTLY AMENDED) The method of claim 8, wherein said parameter is a time value indicating when said particular mock resource was last updated.

10. (CURRENTLY AMENDED) The method of claim 1, wherein examining said limited ~~mock~~ environment comprises:

determining whether said particular mock resource has been accessed.

11. (CURRENTLY AMENDED) The method of claim 10, wherein said particular mock resource contains one or more sets of content, wherein said untrusted program executes in a particular portion of memory, and wherein determining whether said particular mock resource has been accessed comprises:

searching said particular portion of said memory for at least one of said one or more sets of content.

12. (ORIGINAL) The method of claim 1, further comprising:

providing information indicating behavior exhibited by said untrusted program.

13. (ORIGINAL) The method of claim 12, wherein said information comprises indications of undesirable behavior exhibited by said untrusted program.

14. (ORIGINAL) The method of claim 1, further comprising:

determining whether said untrusted program has exhibited undesirable behavior; and

in response to a determination that said untrusted program has exhibited undesirable behavior, taking corrective action.

15. (ORIGINAL) The method of claim 14, wherein taking corrective action comprises:

deleting said untrusted program.

16. (ORIGINAL) The method of claim 14, wherein taking corrective action comprises:

providing a warning to a user.

17. (CURRENTLY AMENDED) A computer readable medium comprising instructions which, when executed by one or more processors, cause the one or more processors to execute an untrusted program, said computer readable medium comprising:

establishing a limited environment within a general environment, wherein said limited environment comprises ~~at least~~ one or more mock resources ~~resource~~, wherein said general environment comprises ~~at least~~ one or more real resources ~~resource~~, ~~and~~ wherein said limited environment and said general environment are both ~~implemented using~~ provided by the same ~~type of~~ operating system, and wherein programs executing within said limited environment cannot access the one or more real resources in said general environment;

executing at least a portion of an untrusted program within said limited environment; and

examining said limited environment after execution of at least said portion of said untrusted program to check for undesirable behavior exhibited by said untrusted program.

18. (CANCELLED).

19. (ORIGINAL) The computer readable medium of claim 17, wherein said limited environment comprises a shell in a UNIX operating system environment.

20. (CURRENTLY AMENDED) The computer readable medium of claim 17, wherein said instructions for causing one or more processors to examine said limited ~~mock~~ environment comprises:

instructions for causing one or more processors to determine whether ~~said~~ a particular mock resource of said one or more mock resources has been deleted.

21. (CURRENTLY AMENDED) The computer readable medium of claim 17, wherein said instructions for causing one or more processors to examine said limited ~~mock~~ environment comprises:

instructions for causing one or more processors to determine whether ~~said~~ a particular mock resource of said one or more mock resources has been renamed.

22. (CURRENTLY AMENDED) The computer readable medium of claim 17, wherein said instructions for causing one or more processors to examine said limited ~~mock~~ environment comprises:

instructions for causing one or more processors to determine whether ~~said~~ a particular mock resource of said one or more mock resources has been moved.

23. (CURRENTLY AMENDED) The computer readable medium of claim 17, wherein said instructions for causing one or more processors to examine said limited ~~mock~~ environment comprises:

instructions for causing one or more processors to determine whether ~~said~~ a particular mock resource of said one or more mock resources has been altered.

24. (CURRENTLY AMENDED) The computer readable medium of claim 23, wherein said particular mock resource has a parameter associated therewith which changes when said particular mock resource is altered, and wherein said instructions for causing one or more processors to determine whether said particular mock resource has been altered, comprises:

instructions for causing one or more processors to determine whether said parameter has changed.

25. (CURRENTLY AMENDED) The computer readable medium of claim 24, wherein said parameter is a time value indicating when said particular mock resource was last updated.

26. (CURRENTLY AMENDED) The computer readable medium of claim 17, wherein said instructions for causing one or more processors to examine said limited ~~mock~~ environment comprises:

instructions for causing one or more processors to determine whether said particular mock resource has been accessed.

27. (CURRENTLY AMENDED) The computer readable medium of claim 26, wherein said particular mock resource contains one or more sets of content, wherein said untrusted program executes in a particular portion of memory, and wherein said instructions for causing one or more processors to determine whether said particular mock resource has been accessed comprises:

instructions for causing one or more processors to search said particular portion of said memory for at least one of said one or more sets of content.

28. (ORIGINAL) The computer readable medium of claim 17, further comprising:

instructions for causing one or more processors to provide information indicating behavior exhibited by said untrusted program.

29. (ORIGINAL) The computer readable medium of claim 28, wherein said information comprises indications of undesirable behavior exhibited by said untrusted program.

30. (ORIGINAL) The computer readable medium of claim 17, further comprising:

instructions for causing one or more processors to determine whether said untrusted program has exhibited undesirable behavior; and

instructions for causing one or more processors to, in response to a determination that said untrusted program has exhibited undesirable behavior, take corrective action.

31. (ORIGINAL) The computer readable medium of claim 30, wherein said instructions for causing one or more processors to take corrective action comprises:

instructions for causing one or more processors to delete said untrusted program.

32. (ORIGINAL) The computer readable medium of claim 30, wherein said instructions for causing one or more processors to take corrective action comprises:

instructions for causing one or more processors to provide a warning to a user.

33. (CANCELLED).

34. (CANCELLED).

35. (CANCELLED).

36. (CANCELLED).

REMARKS

Reconsideration of the application in view of the above amendments and the following remarks is respectfully requested. Claims 1, 4-11, 17, and 20-27 have been amended. No claims have been added. Claims 2, 18, and 33-36 have been cancelled. Thus, Claims 1, 3-17, and 19-32 are currently pending in the application.

Interview Summary

The Applicant thanks the Examiner for the Interview conducted on March 21, 2005. The interview was between Examiner Wu and the Applicant's Attorney, Christopher J. Brokaw. Pending Claim 1 that was rejected in the Office Action was discussed along with U.S. Patent No. 5,842,002 issued to Schnurer et al. ("*Schnurer*"). No agreement was reached. The Applicant is providing herein the amendment that was discussed during the interview.

Claim Rejections – 35 U.S.C. § 103(a)

In the Office Action, the Examiner rejected Claims 1-36 under 35 U.S.C. § 103(a) as being anticipated by *Schnurer*. Claims 1 and 17 have been amended to more particularly identify and distinctly claim subject matter to which the Applicant wishes to receive patent protection.

Independent Claim 1

With regard to independent Claim 1, there is recited:

> A computer-implemented method for generating a transformation document, comprising:

> establishing a limited environment within a general environment, wherein said limited environment comprises at least one mock resource, wherein said general environment comprises at least one real resource, wherein said limited environment and said general environment are both provided by the same operating system, and wherein programs executing within said limited environment cannot access the one or more real resources in said general environment;
>
> executing at least a portion of an untrusted program within said limited environment; and
>
> examining said limited environment after execution of at least said portion of said untrusted program to check for undesirable behavior exhibited by said untrusted program (emphasis added).

Claim 1 provides an advantageous method for executing an untrusted program. According to Claim 1, a computer-implemented method establishes a limited environment within a general environment. The general environment comprises one or more real resources, while the limited environment comprises one or more mock resources. The general environment and the limited environment are both provided by the same operating system. Programs executed within the limited environment cannot access the one or more real resources of the general environment. The limited environment is examined after execution of the untrusted program to check for undesirable behavior exhibited by the untrusted program. Advantageously, the behavior of the untrusted program may be verified without putting the real resources in the general environment at risk.

Such a method is neither disclosed nor suggested by *Schnurer*. Instead, *Schnurer* discloses an approach for a computer virus trapping device ("the trapping device") that creates a virtual world that simulates a host computer system ("the protected device") intended by a virus to infect (Abstract). To prevent the virus from spreading from the trapping device to other devices when the trapping device accesses suspicious data that may contain a virus, the trapping device includes an emulation means that emulates an operating environment (i.e., the "virtual world") that is foreign to the native operating system of the trapping device (Col. 7, lines 4-18). In other words, to ensure that viruses cannot spread from the trapping device, (a) the virtual world simulates the operating system of the protected device, and (b) the trapping device uses a different operating system than emulated by the virtual world. The trapping device is placed in front of the protected device. The trapping device passes suspicious data containing the potential virus directly through to the protected device, in addition to simultaneously processing the data containing the potential virus. (Col. 6, lines 57-58; trap device 10 in FIG. 3 and FIG. 4; Col. 8, lines 50-52). Upon detection of a virus, the trapping device may take an action, such as notifying an administrator (See step 52 of FIG. 1).

At a high level, both *Schnurer* and Claim 1 are directed towards similar subject matter, namely executing a potential computer virus in an emulated environment. However, beyond this broad generalization, the approach of *Schnurer* and the approach of Claim 1 are fundamentally different.

Given the stark contrasts between *Schnurer* and the approach of Claim 1, *Schnurer* cannot possibly teach a system wherein a general environment as claimed exists on either (a) the protected device, or (b) the trapping device. *Schnurer* teaches a method in which a virtual world is emulated on a different physical machine than the physical machine to be protected (Col. 6, lines 57-58; trap device 10 in FIG. 3 and FIG. 4; Col. 8, lines 50-52), thus *Schnurer* cannot possibly teach a general environment, as featured in Claim 1, existing on the protected device because, as featured in Claim 1, the general environment and the limited environment are provided by the same operating system.

Also, *Schnurer* teaches that to prevent a virus from spreading from the trapping device when the trapping device accesses suspicious data, the trapping device includes an emulation means that emulates an operating environment that is foreign to the operating system of the trapping device (Col. 7, lines 4-18). As a result, *Schnurer* cannot possibly teach a system wherein a general environment, as featured in Claim 1, exists on the trapping device because, as featured in Claim 1, the limited environment and the general environment are provided by the same operating system. These and other fundamental differences between the approach of the claims and that of *Schnurer* shall be explained in further detail below.

Schnurer Does Not Teach the General Environment on the Protected Device

Claim 1 recites the feature of:

> "wherein the limited environment and the general environment are both provided by the same operating system"

In citing *Schnurer* to reject Claim 1, the argument of the Office Action rests on a showing that the emulated, virtual world environment of *Schnurer* is analogous to the limited environment, and the operating system of the protected device is analogous to the general environment.

Assuming, *arguendo*, that such an analogy may be made, numerous elements of Claim 1 would not be satisfied by *Schnurer*.

Schnurer shows an emulated environment (generated by emulation means 48 on virus trapping device 10) that is completely separate from the general environment to be protected. *Schnurer* makes clear that virus trapping device 10 is separate from the device to be protected. *Schnurer* states:

> the file server 42 is the computer system to be protected. The virus trapping device 10 is placed in the data stream that connects the filer [sic] server 42 to other workstations 38…In this scenario, all traffic to and from the file server 42 is monitored for viruses by the trap 10 (Col. 6, lines 42-50).

Further, the figures of *Schnurer* clearly identify the trapping device 10 to be different from the computer system to be protected. For example:

(1) On FIG. 1, the emulation box 48 is separate and distant from the protected computer system 28;

(2) On FIG. 3, the trapping device 10 is separate and distinct from the device to be protected 42; and

(3) On FIG. 4, the trapping device 10 is separate and distinct from the node to be protected 32.

Each physical machine executes its own operating system. Since the trapping device and the protected device are clearly different machines, the trapping device and the protected device are executing different operating systems. Thus, the approach of *Schnurer* cannot possibly show "wherein the limited environment and the general environment are both provided by the same operating system" as featured in Claim 1. Therefore, it is respectfully submitted that this element is not disclosed, taught, or suggested by *Schnurer*, and that Claim 1 is patentable over the cited art and is in condition for allowance.

Moreover, as explained below, an argument that *Schnurer* teaches the general environment, as featured in Claim 1, by the trapping device cannot be supported.

Schnurer Does Not Teach a General Environment on the Trapping Device

Claim 1 recites the features of:

> "wherein said limited environment and said general environment are both provided by the same operating system,"
>
> and
>
> "wherein programs executing within said limited environment cannot access the one or more real resources in said general environment;"

As shown in the above-quoted elements of Claim 1, Claim 1 requires that (a) the limited and the general environment are provided by the same operating system, and (b) programs executing within the limited environment cannot access the one or more real resources of the general environment. *Schnurer* provides no teachings of these features.

In sharp contrast, *Schnurer* contains very strong language indicating that the approach of *Schnurer* can only ensure that the virus may be contained within the virtual world of *Schnurer* when the virtual world is emulated using a different operating system that the operating system of the trapping device creating the virtual world.

For example, *Schnurer* states:

> "without the use of a foreign operating system the invention itself risks contamination. A foreign operating system different from the one being protected is introduced into the data stream before the data arrives at the computer system to be protected" (Col. 4, lines 16-20).
>
> "the virus cannot escape the emulation box 48 because the box exists in a foreign operating environment…" (Col. 7, lines 15-16).
>
> "One characteristic of almost all viruses is that one their own they are not capable of crossing from one computer OS to another" (Col. 3, lines 59-61).

Thus, it is clear that the approach of *Schnurer* requires the emulated, virtual world to be implemented using a different type of operation system than the trapping device to prevent viruses from escaping from the trapping device.

While *Schnurer* does raise the possibility that the emulated, virtual world may emulate the same type of operating system as used by the trapping device, *Schnurer* is clear that in doing so viruses may escape the trapping device. For example, *Schnurer* states:

> "The inventors recognize that it can be done without a transplatform, but it will be slow and absolutely unsafe. The use of a foreign operating system can be likened to the use of lead walls and glass walls and mechanical arms used by people manipulating radioactive materials in a lab. While it is certainly possible to pick up radioactivity with one's bare hands, it is not highly recommended or is it safe. While the invention can be had without the use of a foreign operating system, it is not highly recommended nor is it safe" (Col. 4, line 63 – Col. 5, line 5).

> "The use of a foreign operating system guarantees the invention a high degree of safety and impenetrability. While the inventors recognize that such invention can be built without the use of a foreign operating system, such a version of the invention would lack any credible degree of security. In addition, without the use of a foreign operating system the invention itself risks contamination." (Col. 4, lines 11-17).

Thus, to the extent that *Schnurer* teaches using the same type of operating system in the virtual world as the operating system of the trapping device, *Schnurer* does not teach any way that such an approach could satisfy the limitation of "wherein programs executing within said limited environment cannot access the one or more real resources in said general environment" as expressly featured in Claim 1.

Further, *Schnurer* does not contain any suggestion that the emulated, virtual world is the same operating system as that used by the trapping device. Indeed, *Schnurer* teaches away from such a suggestion, since that characterization of an "emulated, virtual world" clearly is different and distinct from the real operating system employed by the trapping device. As a result, the approach of *Schnurer* cannot possibly show the above-quoted combination of elements featured in Claim 1.

As one or more express elements of Claim 1 are not disclosed, taught, or suggested by *Schnurer*, it is respectfully submitted that Claim 1 is patentable over the cited art and is in condition for allowance.

Claims 3-17 and 19-32

Claims 3-16 are dependent claims, each of which depends (directly or indirectly) on Claim 1. Each of Claims 3-16 is therefore allowable for at least the reasons given above with respect to Claim 1. In addition, each of Claims 3-16 introduces one or more additional limitations that independently render it patentable. Due to the fundamental differences already identified, to expedite the positive resolution of this case, a separate discussion of the limitations of Claims 3-16 is not included at this time. The Applicant reserves the right to further point out the differences between the cited art and the novel features recited in the dependent claims.

Claims 17 and 19-32 include limitations similar to Claims 1 and 3-16 respectively, except in the context of computer-readable media. It is therefore respectfully submitted that Claims 17 and 19-32 are patentable over *Schnurer* for at least the reasons given above with respect to Claims 1 and 3-16.

A Java Virtual Machine is Not a Limited Environment

During the Interview of March 21, 2005, a suggestion was made that a Java Virtual Machine (JVM) may be analogous to a limited environment. The Applicant respectfully disagrees that a JVM could suggest a limited environment as claimed, due to the numerous express claim limitations that would not be met. For example, Claim 1 recites the features of "wherein said limited environment comprises one or more mock resources," "wherein said limited environment and said general environment are both provided by the same operating system," and "wherein programs executing within said limited environment cannot access the one or more real resources in said general environment." It is respectfully submitted that a JVM (a) contains real resources, not mock resources, (b) is not provided by the same operating system as a general environment, and (c) programs executed within a JVM can access the real resources of a general environment, e.g., the motivation of providing a JVM is to enable programs to access the real resources of the machine executing the JVM.

As a result, a JVM cannot possible be analogous to a "limited operating system environment," as featured in Claim 1.

CONCLUSION

For the reasons given above, the Applicant submits that the pending claims are patentable over the art of record, including the art cited but not applied. Accordingly, allowance of all pending claims is respectfully solicited.

The Examiner is invited to telephone the undersigned at (408) 414-1225 to discuss any issue that may advance prosecution.

No fee is believed to be due specifically in connection with this Reply. The Commissioner is authorized to charge any fee that may be due in connection with this Reply to our Deposit Account No. 50-1302.

Respectfully submitted,

HICKMAN PALERMO TRUONG & BECKER LLP

Dated: April 8, 2005

Christopher J. Brokaw
Reg. No. 45,620

2055 Gateway Place, Suite 550
San Jose, California 95110-1089
Telephone No.: (408) 414-1225
Facsimile No.: (408) 414-1076

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

On April 8, 2005 By Angelica Maloney
Angelica Maloney